# Mise à jour d'un projet Symfony

Au moment où j'écris cette page, nous sommes en train de mettre à jour tous les projets du GInfo vers des version plus récentes.

| Date | Symfony | PHP | Composer | Node |
|---|---|---|---|---|
| Age de Rom - 2022 | 4.4 | 7.2 | 1 | 12 |
| 2022 - ? | 5.4 | 8 | 2 | 16 |

Pensez à faire vos modifications sur cette branche au risque de casser tout le projet !

## 1. La partie Back-end

Commencer par supprimmer le fichier `composer.lock` ainsi que les dossiers `vendor` et `var`.

Vérifier que vous avez la bonne versions de PHP et Composer !

L'idée est de mettre à jour tous les packages de composer pour qu'ils soient viables avec les nouvelles versions. Dans le `composer.json`, tous les packages en `symfony/nom_package: 4.4.*` devront être en 5.4. Pour cela nous allons lancer la commande suivante :

```
  composer update "symfony/*"
```

Normalement tous les paquets se mettront à jour avec la LTS actuelle de Symfony.

Suite à cela voici une liste des petits problèmes que nous avons rencontré, la liste sera à l'avenir complétée pour documenter notre travail :) :

### doctrine_migrations.yaml

Il faut modifier le fichier `config/packages/doctrine_migrations.yaml`. On passe de ça:

```
doctrine_migrations:
    dir_name: '%kernel.project_dir%/src/Migrations'
    # namespace is arbitrary but should be different from App\Migrations
    # as migrations classes should NOT be autoloaded
    namespace: DoctrineMigrations
```

à ça:

```
doctrine_migrations:
    migrations_paths:
        'DoctrineMigrations': '%kernel.project_dir%/src/Migrations'
```

```
    enable_profiler: '%kernel.debug%'
```

## Récupérer le repository lié à une classe

Avant:

```
$data = $this->getDoctrine()->getRepository('App:MyEntity')->getData($id);
```

Après:

```
$data = $this->getDoctrine()->getRepository(MyEntity::class)->getData($id);
```

## Logout CAS

On a eu besoin d'ajouter une route de logout.

```
/**
 * @Route("/logout", name="logout")
 */
public function logoutAction(ParameterBagInterface $param) {
    if (($param->get('cas_logout_target') !== null) &&
(!empty($param->get('cas_logout_target')))) {
\phpCAS::logoutWithRedirectService($param->get('cas_logout_target'));
    } else {
        \phpCAS::logout();
    }
}
```

## UserInterface

Attention, si votre classe User implémente UserInterface alors la méthode `getUsername` est nécessaire. (même si vous ne nous en servez pas)

## ManagerRegistry

Avant:

```
use Doctrine\Common\Persistence\ManagerRegistry;
```

Après:

```
use Doctrine\Persistence\ManagerRegistry;
```

### Listener (CASListener dans notre cas)

https://stackoverflow.com/questions/56438177/deprecated-the-listenerinterface-turn-your-listeners-into-callables-instead

Avant:

```
use Symfony\Component\HttpKernel\Event\GetResponseEvent;
use Symfony\Component\Security\Http\Firewall\ListenerInterface;

class CasListener implements ListenerInterface {
    public function handle(GetResponseEvent $event) {
        ...
    }
    public function checkHandleLogout(GetResponseEvent $event) {
        ...
    }
}
```

Après:

```
use Symfony\Component\HttpKernel\Event\RequestEvent;

class CasListener {
    public function __invoke(RequestEvent $event) {
        ...
    }
    public function checkHandleLogout(RequestEvent $event) {
        ...
    }
}
```

### PHPUnit

A rajouter dans le `.gitignore` Pour éviter les dingueries:

```
###> phpunit/phpunit ###
/phpunit.xml
.phpunit.result.cache
###< phpunit/phpunit ###
```

## 2. La partie Front-end

Commencez par supprimer le fichier `package-lock.json` et le dossier `node_modules`.

Vérifier que vous avez la bonne version de Node. *(nvm tout ça tmtc)*

Lancer un gros `npm update` sa mère et corriger les éventuels bugs qui pourrait apparaitre.

Attention à ne pas utiliser des versions trop récentes si certains packages ont des dépendances trop anciennes (mercé AdminBSB)

From:
https://wiki.centrale-med.fr/ginfo/ - **Wiki GInfo**

Permanent link:
**https://wiki.centrale-med.fr/ginfo/formations:symfony_maj**

Last update: **01/06/2022 22:41**